Numerické programovanie aj pre TMF

Užitočné rady ako začať, a kde sa inšpirovať

Matej Badin

11. novembra 2025

Motivácia

TMF je (aj) dobrý priestor pre rozvíjanie programovania. Vlastný program nám vie mnohé veci uľahčiť (*analýza exp. dát, zautomatizovanie experimentu*) alebo sprístupniť (*simulácia fyz. javu z prvých princípov*).

Programovanie a simulácie nám popri analytickej teórii a experimentoch ponúkajú tretí spôsob ako možno vedecký problem uchopiť a skúmať (*simulácia = virtuálny experiment v počítači*).

Schopnosť vedieť písať vlastný high-performance kód sa môže ukázať, ako kľúčová zručnosť navyše popri štúdiu na vysokej škole alebo vedeckej práci.

- "*Dôveruj, ale preveruj!*"
  Ivan Košinár, Úvodné sústredenie pre študentov fyziky na FMFI; v roku 2014
- "*The first principle is that you must not fool yourself, and you are the easiest person to fool.* Richard P. Feynman
- Pre numerické riešenie dif. rovníc viď prezentáciu Jakuba Šťavinu z US TMF 2024.
- "*Učíme sa napokon všetko sami.*¨
  Dobré zdroje a učitelia nám vedia tento proces urýchliť. Táto prezentácia odkazuje na (subjektívny) výber, čo autor považuje za dobré zdroje na samoštúdium.
- "*Niekde začať treba.*"
  Zdroje sú usporiadané podľa náročnosti a dostupnosti zdrojov od základných kurzov k programovaniu až po pokročilé témy.
- Existuje už veľa dobrých zdrojov, cieľom tejto prezentácie je ich zoskupiť, nie ich nahradiť. A vybrať tie v akom poradí sa na čo sústrediť.

Obsah

- Základy - C++ a Python
- Prehľad štandardných problémov a dátových štruktúr
- Tipy pre numerické programovanie
- Numerické a iné knižnice
- Paralelné programovanie
  "*Moderné procesory majú viac jadier. Ako to využiť?*"
- Programovanie pre GPU
  "*Počítač nie je na hranie ...*"
- Kde sa inšpirovať?

Python alebo C++? A iné otázky

- Rôzne programovacie jazyky s rôznou pôsobnosťou využitia.
- Na TMF nechceme znovu vymýšlať koleso. Písať vlastný integrátor diferenciálnych rovníc v C++ nechcete (existujú hotové knižnice pre Python[1]) a/alebo FEM simuláciu prúdenia kvapaliny tiež nie[2]. Viď prezentáciu Jakuba Šťavinu.
- Času pre TMF je mimo iných školských povinností málo, tak ho chcete investovať efektívne.
- Je viac ako pravdepodobné, že mnoho z tejto prezentácie priamo na TMF nevyužijete, môže to ale poslúžiť ako vhodný odrazový mostík, ak Vás fyzika nadchla a študujete ju aj na vysokej škole.

[1] Napísané mimochodom v C++.

[2] Pri všetkej úcte, asi je to mimo Vašich časových možností

C++

- Kompilovaný kód t.j. pred spustením je kód je kompilovaný do inštukčnej sady procesora počítača. Následne už môžeme spúšťať iba skompilovaný kód.
- Široké uplatnenie - od kávovaru po raketu vrátane operačného systému Vášho počítača. T.j. je veľmi používaný = opdframe
  latí sa ho naučiť.
- Stále vyvíjaný - rôzne štandardy C++03, C++11, C++14, C++17, ...
- Veľmi obľúbený aj vo vedeckej komunite, pretože je modernejší ako Fortran, multiparadigmový a ak je kód správne napísaný, tak aj rýchly (*programovanie aplikácií pre superpočítače = high-performance computing*).

Kde sa naučiť C++?

Online zdroje:

- C++ Reference - `en.cppreference.com`
- Learn C++ - `learncpp.com`
- Kurz 22 prednášok od C++03 po C++23
  Federico Busato - Modern C++
  `github.com/federico-busato/Modern-CPP-Programming`

Knižky:

- S. Prata. C++ Primer Plus
- B. Stroustrup. A Tour Of C++

Objektovo-orientované programovanie

C++ je (aj) objektovo-orientovaný jazyk, čo nám umožňuje mať hrierchiu objektov, ktoré môžu navzájom dediť rôzne vlastnosti alebo funkcionalitu. To umožňuje napr. rozširovať logiku kódu už existujúcich knižníc, elegantne rozširovať kód, oddeliť implementáciu algoritmu od rozhrania objektu, minimalizovať programovanie prístupom `CTRL+C & CTRL+V` a pod. Mnohé situácie už majú svoj idióm.

Oplatí sa pozrieť:

- tzv. Design Patterns - štandardné situácie pri používaní OOP
- `refactoring.guru`
- `en.wikipedia.org/Solid`
- `en.wikibooks.org/wiki/More_C++_Idioms`

C++ - Špecializované knižky

Verzia C++11 priniesla tzv. move semantics, t.j. nové možnosti ako možno prenášať obsah jedného objektu do inej inštancie toho istého objektu. Je o tom aj celá knižka :)

- N. M. Josuttis. C++ Move Semantics

Druhou častou používanou možnosťou sú tzv. templates, ktoré umožňujú napr. generovať rôzne verzie výpočtového kernelu pre rôzne parametre ale aj písať kód agnosticky voči typu premennej objektu, s ktorým pracujeme. Tiež je o tom aj celá knižka, ale postačia aj predchádzajúce referencie a náhľad už do existujúceho nejakého fyzikálneho kódu.

- D. Vandevoorde. C++ Templates: The Complete Guide

C++ - Špecializované knižky

Zbierky tipov a trikov:

- S. Meyers. Effective C++ 55 Specific Ways to Improve Your Programs and Designs
- S. Meyers. Effective Modern C++ 42 Specific Ways to Improve Your Use of C++11 and C++14
- S. Meyers. Effective STL 50 Specific Ways to Improve Your Use of the Standard Template Library
- H. Sutter. Exceptional C++
- H. Sutter. Exceptional C++ Style
- H. Sutter. More Exceptional C++
- ...

Optimalizácia

Kód môžeme optimalizovať na rôznych úrovniach. Najvyššou úrovňou je vymyslenie nového algoritmu alebo použitie dátovej štruktúry, ktorá nám zlepší napr. asymptotickú časovú zložitosť z kubickej na kvadratickú ($\mathcal{O}(N^3) \rightarrow \mathcal{O}(N^2)$).

Iný druh optimalizácie je napríklad optimalizácia pomeru počtu prístupu do pamäti k počtu výpočtových krokov algoritmu vnútri časti algoritmu, ktorý sa často opakuje. V závislosti od tohto pomeru ku architektúre počítača môže byť kód limitovaný rýchlosťou procesora alebo rýchlosťou prístupu do pamäte.

Iný druh optimalizácie využíva tzv. *vektorizáciu* t.j. inštrukčnú sadu procesora známu pod skratkami ako `AVX`, `AVX-512`, ... (**SIMD** (single instruction - multiple data) paradigma).

- K. Guntheroth. Optimized C++: Proven Techniques for Heightened Performance
- D. Kusswurm. Modern Parallel Programming with C++ and Assembly Language x86

Python

- Interpretovaný jazyk, vhodný na analýzu dát, kreslenie grafov, automatizovanie exp. aparatúry, ...
- Referencia - `python.org`
- Bohatá štandardná knižnica
- Bohatý ekosystém knižníc a ich správy - *Poetry, Numpy, Scipy, Matplotlib, Numba*, ...
- ďalšie tzv. *domain specific languages* postavené na Python-e.

Môže sa tiež hodiť

Pochopiť niečo o tom ako fungujú počítače a operačný systém:

- A. S. Tanenbaum a H. Bos. Modern Operating Systems
- R. a A. Arpaci-Dusseau. Operating Systems. Three Easy Pieces
- A. Silberschatz et al. Operating System Concepts

Pochopiť ako fungujú počítačové siete:

- A. S. Tanenbaum. Computer Networks

niečo málo o kryptografii[3]:

- Katz a Lindell. Introduction to Modern Cryptography.
- J. P. Aumasson. Serious Cryptography: A Practical Introduction to Modern Encryption.

niečo málo o kompresii:

- D. Salomon. Data Compression

[3] Viď archív IYPT úloh. Už sa vyskytli úlohy, kde sa vyslovene žiadalo použiť koncepty z kryptografie.

Prehľad štandardných problémov a dátových štruktúr

Prehľad štandardných problémov a dátových štruktúr

Častokrát môžeme ako súčasť väčšieho problému riešiť nejakú zo štandardných úloh, alebo naopak pre naše účely sa môže hodiť nejaká zo štandardných dátových štruktúr.

- **Algorithms for Competitive Programming** - `cp-algorithms.com` - stránka primárne určená pre záujemcov o súťaže typu Olympiáda v informatike, ale obsahuje veľmi obšírny prehľad rôznych dátových štruktúr a algoritmov.

Ďalšie užitočné lokálne zdroje:

- **Modrá a zelená zbierka úloh KSP** - zbierka úloh, ktorá obsahuje úlohy zatriedené do rôznych kategórii podľa druhu algoritmu využívanom pri jej riešení.
- **Kuchárka KSP** - prehĽad algoritmov a dátových štruktúr.
- **Pruvodce labyrintem algoritmu** - prehľad algoritmov a dátových štruktúr.

Tipy pre numerické programovanie

Tipy pre numerické programovanie

High-perfomance vedecké kódy, používajú naraz niekoľko levelov optimalizácie. Od voľby vhodných algoritmov po ich vhodnú implementáciu, kde napríklad nepočítame niečo viackrát ako naozaj musíme. Dobrým zdrojom rôznych tipov a trikov, ktoré sa používajú sú väčšinou tréningové materiály rôznych výpočtových centier.

Písanie vlastnej simulácie k TMF úlohe, tak môže byť aj vhodným pieskoviskom pre pestovanie tejto zručnosti.

Užitočné zdroje:
- `fz-juelich.de/en/jsc/education/training-courses/training-materials`

Numerické a iné knižnice

Numerické a iné knižnice

Oplatí sa nevymýšlať znovu koleso. Pri TMF to platí viacnásobne! Platí to aj pri programovaní, častokrát existuje už solídna alebo často používaná knižnica. Užitočné knižnice (aj pre C++):

- **BLAS** - knižnica pre skalárne, vektorové a maticové operácia (backend mnohých numerických knižníc).
- **LAPACK** - knižnica pre algoritmy z lineárnej algebry.[4]
- **ScaLAPACK** - LAPACK pre sparse problémy t.j. problémy obsahujúce riedke matice.
- **OpenCV** - knižnica pre počítačové spracovanie obrazu.
- **Qt** - knižnica pre tvorbu grafických rozhraní.

[4] Časť matematiky, ktorá sa učí až na vysokej škole.

Paralelné programovanie

Paralelné programovanie

Je svet sám o sebe. Keď vektorizácia nestačila a zmenšovanie litografického procesu už neviedlo k zvyšovaniu frekvencie čipov, tak výkon superpočítačov začal rásť zväčšovaním počtu jadier (cores) a zapájaním do výpočtu celých skupín procesorov (nodes).

Štandardne sa rôzne levely paralelizácie riešia cez rôzne frameworky, napr. paralelizácia medzi viacero jadier na tom istom počítači (node) pomocou **OpenMP** alebo **C++ threads** (vlákien), kdežto komunikácia počítačov tvoriacich cluster prebieha počas výpočtu cez framework **MPI**.

Užitočné zdroje:

- An Introduction to Parallel Programming
- Eijkhout - Introduction to HPSC
- Eijkhout - Parallel Programming Using MPI and OpenMP
- Parallel and High Performance Computing
- Quinn Parallel Programming in C with MPI and OpenMP
- Structured Parallel Programming Patterns for Efficient Computation

C++ threads

Ponúkajú low level prístup ako písať aplikácie bežiace na jednom počítači ale využívajúcich viac vlákien operačného systému, a teda aj potenciálne jadier procesora.

Užitočné zdroje:

- Breshears C - The Art of Concurrency A Thread Monkey's Guide to Writing Parallel Applications
- C++ Concurrency in Action

OpenMP

Prístupnejší higher level framework dekorujúci kód cez (`# pragma`) syntax využívajúci vo svojom vnútri C++ threads, tiež zacielený pre využitie viacerých jadier procesora.

Užitočné zdroje:

- OpenMP 5 Reference
- OpenMP Cheat sheet
- Parallel Programming in OpenMP
- Using OpenMP - The Next Step
- Using OpenMP

MPI

MPI (**message passing interface**) je framework umožňujúci písať aplikácie využívajúcich viacero jadier resp. umožňuje písať aplikácie bežiace aj na *clustri* (sieť počítačov, kde aplikácia beží distribuovaná na viacerých počítačoch súčasne). Je to pomerne starý (1993), ale stále sa vyvíjajúci sa štandard. Aplikácie napísané cez MPI sa nespúšťajú ako iné aplikácie, ale musia sa spustiť cez špeciálny runner zabezpečujúci spustanie aplikácie na viacerých počítačoch.

Užitočné zdroje:

- MPI The Complete Reference
- Parallel programming with MPI
- Parallel Scientific Computing in C and MPI
- Using Advanced MPI Modern Features of the MPI
- Using MPI Portable Parallel Programming with the MPI
- Using MPI 2 Advanced Features

Programovanie pre GPU

Programovanie pre GPU

V roku 2001 vznikli špeciálne (programovateľné) akcelerátory pre uľahčenie matematických výpočtov vyskytujúcich sa v počítačovej grafike. Neskôr si vedci uvedomili, že tieto akcelerátory (grafické karty) možno využiť aj na iné druhy výpočtov, napr. vedecké, kde sa často opakuje tá istá inštrukcia, ale aplikovaná na rôzne dáta - príklad *numerické riešenie parciálnych diferenciálnych rovníc*. Ide o tzv. **SIMT** (single instruction - multiple threads) paradigmu, `threads` sú organizované do skupín tzv. warpov, ktoré vykonávajú tú istu sadu inštrukcií.

Pre programovanie sa využívajú hlavne tieto frameworky:

- CUDA
- OpenACC
- OpenCL, ale aj iné.

Veľmi živá je aj

- GPU MODE komunita na Discorde.

CUDA

Najrozšírenejší framework spustiteľný na grafických kartách od firmy NVIDIA, vrátane veľkého množstva podporných knižníc a nástrojov ako napr. cuBLAS, a pod.

Užitočné zdroje:

- Programming Massively Parallel Processors A Hands-on Approach
- CUDA Reference
- kurzy z FSZ Juelich

OpenACC

Podobný higher level framework dekorujúci kód cez (`# pragma acc`) syntax ako OpenMP. Primárne cielené pre rôzne akcelerátory ako grafické karty. Spustiteľný aj na grafických kartách od iných firiem ako NVIDIA.

Užitočné zdroje:

- OpenACC 3 Reference
- OpenACC for Programmers Concepts and Strategies
- OpenACC Programming Guide
- Parallel Programming with OpenACC

OpenCL

Famework spustiteľný na grafických kartách od rôznych výrobcov.

Užitočné zdroje:

- Heterogeneous Computing with OpenCL
- OpenCL in Action
- OpenCL Programming Guide
- OpenCL Programming by Exampe
- Using OpenCL Programming Massively Parallel

Kde sa inšpirovať?

Kde sa inšpirovať ako písať a organizovať vedecký kód

najlepšie v nejakom už existujúcom väčšom open-source projekte, napr.:

- OpenCV - knižnica na počítačové spracovanie obrazu.
- Python - Verzia Pythonu implementovaná v C++.
- LAMMPS - Open-source kód pre molekulovú dynamiku.
- PLUMED - Plugin pre softvér vyššie implementovaný ako knižnica.
- QE - Vedecký softvér napísaný vo Fortrane.

Oplatí sa preštudovať jednoducho čítaním a prechádzaním cez kód[5], ako je cudzí kód organizovaný, ako sa využíva existencia statických a dynamických knižníc pre rozširovanie modularity kódu a pod.

Pozor! To, že je niečo open-source ešte neznamená, že to nemôže obsahovať chyby. Alebo, že nevymyslíte niečo lepšie.

[5] Vyžaduje si to však dostatočný časový vklad čitateľa

Ďakujem za pozornosť!